Motion planning with pulley, rope, and baskets*

Christian E.J. Eggermont[1] and Gerhard J. Woeginger[1]

1 **Department of Mathematics and Computer Science, TU Eindhoven, P.O. Box 513, 5600 MB Eindhoven, Netherlands**

Abstract

We study a motion planning problem where items have to be transported from the top room of a tower to the bottom of the tower, while simultaneously other items have to be transported into the opposite direction. Item sets are moved in two baskets hanging on a rope and pulley. To guarantee stability of the system, the weight difference between the contents of the two baskets must always stay below a given threshold.

We prove that it is Π_2^p-complete to decide whether some given initial situation of the underlying discrete system can lead to a given goal situation. Furthermore we identify several polynomially solvable special cases of this reachability problem, and we also settle the computational complexity of a number of related questions.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases planning and scheduling; computational complexity

Digital Object Identifier 10.4230/LIPIcs.STACS.2012.374

1 Introduction

The Oxford mathematician Charles Lutwidge Dodgson (1832–1898) is better known under his pseudonym Lewis Carroll. He is the author of books like *"Alice's Adventures in Wonderland"* and *"Through the Looking-Glass"*, and he has constructed a multitude of mathematical puzzles. One of Carroll's most famous problems is called *"The Captive Queen"*; see for instance Wakeling [10]:

> *"A captive queen and her son and daughter were shut up in the top room of a very high tower. Outside their window was a pulley with a rope around it, and a basket fastened to each end of the rope of equal weight. They managed to escape with the help of this and a weight they found in the room, quite safely. It would have been dangerous for any of them to come down if they weighed 15 lbs more than the content of the other basket, for they would do so too quick, and they also managed not to weigh less either. The one basket coming down would naturally of course draw the other basket up.*
> *The queen weighed 195 lbs, daughter 105, son 90, and the weight 75 lbs. How did they all escape safely?"*

In the initial situation queen, daughter, son, and weight are all up the tower and none of them is at the bottom of the tower. This situation is denoted $Q, D, S, W \parallel \emptyset$, and we use a similarly intuitive notation for other situations. The schedule in Figure 1 solves the Captive Queen problem in eleven steps.

* This research has been supported by the Netherlands Organisation for Scientific Research (NWO), grant 639.033.403, and by DIAMANT (an NWO mathematics cluster)

29th Symposium on Theoretical Aspects of Computer Science (STACS'12).
Editors: Christoph Dürr, Thomas Wilke; pp. 374–383
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1.	The weight is sent down	$Q, D, S \parallel W$
2.	The son goes down, the weight comes up	$Q, D, W \parallel S$
3.	The daughter goes down, and the son comes up	$Q, S, W \parallel D$
4.	The weight goes down	$Q, S \parallel D, W$
5.	The queen goes down; daughter and weight come up	$D, S, W \parallel Q$
6.	The weight falls down	$D, S \parallel Q, W$
7.	The son goes down, the weight comes up	$D, W \parallel Q, S$
8.	The daughter goes down, and the son comes up	$S, W \parallel Q, D$
9.	The son sends down the weight	$S \parallel Q, D, W$
10.	The son goes down, the weight comes up	$W \parallel Q, D, S$
11.	The weight falls to the ground	$\emptyset \parallel Q, D, S, W$

Figure 1 A feasible schedule for Lewis Carroll's Captive Queen problem.

Mathematical formulation

Motivated by the Captive Queen problem, we will investigate the following motion planning problem. Let I be a set of items, and let $w(i)$ be the positive integer weight of item $i \in I$. A *state* of the system is specified by an item set $J \subseteq I$ at the top of the tower (and with the remaining items in $I - J$ located at the bottom of the tower). For $J \subseteq I$ we throughout denote $w(J) = \sum_{j \in J} w(j)$, and as usual we let $w(\emptyset) = 0$. The system can move directly from state $J \subseteq I$ to state $K \subseteq I$ if

$$|w(J \cap (I - K)) - w(K \cap (I - J))| \ \leq \ \Delta, \tag{1}$$

where the positive integer bound Δ specifies the maximum allowed weight difference between the two exchanged subsets in the baskets. We say that state K is *reachable* from state J, if there is a sequence of moves that transforms J into K. Since inequality (1) is symmetric in J and K, reachability is a symmetric relation. The decision version of our mathematical motion planning problem is defined as follows.

Problem: Captive-Queen

Instance: A set I of items; positive integer weights $w(i)$ for $i \in I$; a positive integer bound Δ; two subsets $I_0, I_1 \subseteq I$.

Question: Is the goal state I_1 reachable from the initial state I_0?

Although problem Captive-Queen does not cover all the algorithmic features of Carroll's problem, we think that it does cover the most important ones. Note that the weight of 75 lbs in Carroll's problem constitutes an indestructible item that can fall down with the basket without obeying constraint (1); in our problem formulation, however, there are no such indestructible items.

We will also discuss the following variant of Captive-Queen where the number of moves is a priori bounded by m.

Problem: Captive-Queen-with-few-Moves
Instance: A set I of items; integer weights $w(i)$ for $i \in I$; a bound Δ; two subsets $I_0, I_1 \subseteq I$; an integer bound m that is encoded in unary.
Question: Is there a sequence of at most m moves that transforms the initial state I_0 into the goal state I_1?

The following example illustrates that there exist YES-instances of Captive-Queen for which every feasible schedule has exponential length.

▶ **Example 1.** Consider the item set $I = \{0, 1, \ldots, n-1\}$ with weights $w(i) = 2^i$ for $i \in I$. The difference bound is $\Delta = 1$, the initial state is $I_0 = \emptyset$, and the goal state is $I_1 = I$.

Let $J_1, \ldots, J_{2^n}$ be an enumeration of all 2^n subsets of I in order of increasing weight. By considering the binary representation of $w(J_j)$ and $w(J_{j+1})$, one sees that the system can move from every state J_j to the successor state J_{j+1}. Since $J_1 = I_0$ and $J_{2^n} = I_1$, there consequently exists a sequence of $2^n - 1$ moves that transforms state I_0 into state I_1. Since every move increases the weight of the current state by at most $\Delta = 1$, every feasible schedule must have length at least $2^n - 1$.

Our results

We establish a number of results on the algorithmic and combinatorial behavior of the motion planning problems introduced above. As our main result, we precisely pinpoint the computational complexity of the Captive-Queen problem: it is Π_2^p-complete and hence located at the second level of the polynomial hierarchy (Section 3). The variant Captive-Queen-with-few-Moves turns out to be NP-complete. Next we show that certain natural special cases of Captive-Queen are polynomially solvable:

- the case with super-increasing weight sequences (Section 4.1);
- the case with divisible weight sequences (Section 4.2).

These special cases originate from the literature around the knapsack problem (see for instance the books [6, 4]). In Section 5 we finally characterize the computational complexity of several related algorithmic problems:

- recognizing isolated states in a system;
- deciding whether every state in a system is isolated;
- deciding whether a system contains some isolated state;
- deciding whether all states in a system are reachable from each other.

We also discuss the restriction of these problems to super-increasing weight sequences and to divisible weight sequences.

2 Preliminaries and first observations

We consider some fixed instance of Captive-Queen with item set I, weights $w(i)$, difference bound Δ, and initial state I_0 and goal state I_1. Throughout the paper we will assume without loss of generality that $w(I_0) \leq w(I_1)$ (and otherwise we simply swap I_0 and I_1).

The following (straightforward) lemma provides a concise characterization of the possible moves between states.

▶ **Lemma 2.** *There is a direct move from state $J \subseteq I$ to state $K \subseteq I$ if and only if*

$$|w(J) - w(K)| \;\leq\; \Delta. \tag{2}$$

Proof. This follows since the inequalities in (1) and (2) are equivalent. ◀

The *weight spectrum* $W_1 < W_2 < \cdots < W_k$ of the Captive-Queen instance enumerates the weights of all the subsets of I in increasing order. The *weight spectrum between* $w(I_0)$ *and* $w(I_1)$ is the piece $W_a < \cdots < W_b$ of the weight spectrum starting with $W_a = w(I_0)$ and ending with $W_b = w(I_1)$. The *maximum gap* between two bounds W_a and W_b is the maximum of the values $W_{j+1} - W_j$ taken over $j = a, \ldots, b-1$.

The standard dynamic programming algorithm for the Subset-Sum problem generates (as a by-product) a sorted list of the sums of all subsets of a given set of integers; see for instance Cormen & al [2]. This yields the following.

▶ **Lemma 3.** *The weight spectrum can be computed in pseudo-polynomial time* $O(nW)$*, where* $n = |I|$ *and* $W = \sum_{i \in I} w(i)$*.* ◀

The following observation is an immediate consequence of Lemma 2.

▶ **Corollary 4.** *The following three statements are pairwise equivalent.*

(i) *The goal state* I_1 *is reachable from the initial state* I_0*.*
(ii) *The maximum gap in the weight spectrum between* $w(I_0)$ *and* $w(I_1)$ *is at most* Δ*.*
(iii) *For every integer* V *with* $w(I_0) \leq V < w(I_1)$*, there exists an item set* $J \subseteq I$ *such that* $V < w(J) \leq V + \Delta$*.*

Lemma 3 and Corollary 4.(ii) together imply that Captive-Queen and Captive-Queen-with-few-Moves are solvable in pseudo-polynomial time.

3 Hardness of the Captive-Queen

In this section we will establish Captive-Queen to be Π_2^p-complete and Captive-Queen-with-few-Moves to be NP-complete.

Corollary 4.(iii) shows that the Captive-Queen problem can be rewritten into an equivalent question of the form $\forall x \exists y\, P(x, y)$ where $P(x, y)$ is a Boolean predicate that can be evaluated in polynomial time. The complexity class Π_2^p represents problems of exactly this particular form with a universal quantifier followed by an existential quantifier (see for instance Section 17.2 in Papadimitriou [7]). Hence we derive the following statement.

▶ **Lemma 5.** *Problem* Captive-Queen *lies in* Π_2^p*.* ◀

The main part of this section is dedicated to proving Π_2^p-hardness of the Captive-Queen problem. The proof is done by means of a polynomial time reduction from the following quantified satisfiability problem, which was shown to be Π_2^p-complete by Stockmeyer [9].

Problem: 2-Quantified 3-CNF-Sat

Instance: Two sets $X = \{x_1, \ldots, x_s\}$ and $Y = \{y_1, \ldots, y_s\}$ of Boolean variables. A Boolean formula $\phi(X, Y)$ over $X \cup Y$ in conjunctive normal with clauses $c_1, \ldots, c_t$ where every (disjunctive) clause c_j consists of exactly three literals.

Question: Is $\forall x_1, \ldots, x_s\, \exists y_1, \ldots, y_s\ \phi(X, Y)$ true?

We pick an arbitrary instance of 2-Quantified 3-CNF-Sat, and we will construct a corresponding instance of Captive-Queen from it. In our construction every item weight is specified in terms of its decimal representation, which consists of $3s + 2t$ digits that are partitioned into five parts; see Figure 2 for an illustration.

- The verification-part consists of the t left-most digits in the decimal representation, and the clause-part consists of the t digits immediately to the right of the verification-part. In both parts the jth digit from the right ($1 \leq j \leq t$) is said to correspond to clause c_j.
- The Y-part consists of the next s digits. In this part the ith digit from the right ($1 \leq i \leq s$) corresponds to variable y_i.
- The X-part consists of the next s digits (immediately to the right of the Y-part). The ith digit from the right ($1 \leq i \leq s$) corresponds to the Boolean variable x_i.
- The control-part consists of the remaining s digits in the decimal representation. For technical reasons, we will mainly work with the s lowest bits in the *binary* representation of the control-part (and we will ignore the remaining unused bits). The ith bit from the right ($1 \leq i \leq s$) corresponds to the Boolean variable x_i.

Verification part	**Clause-part**	**Y-part**	**X-part**	**Control part**
$t \ldots\ldots 1$	$t \ldots\ldots 1$	$s \ldots\ldots 1$	$s \ldots\ldots 1$	$s \ldots\ldots 1$

Figure 2 The division of the decimal representations into five parts.

Throughout we will use the term *digits* to specify the decimal representation of the verification-part, clause-part, Y-part, and X-part, and we will use the term *bits* to specify the binary representation of the control-part. Next, let us describe the $4s + 3t + 2$ items in the Captive-Queen instance together with their weights.

- For every literal $\ell \in \{x_i, \overline{x_i}\}$ there is a corresponding X-item $X(\ell)$. The weight of $X(\ell)$ has a 1-digit in the position corresponding to variable x_i in the X-part. If $\ell = x_i$ is unnegated, then there is a 1-bit in the position that corresponds to x_i in the control-part (whereas in case $\ell = \overline{x_i}$ is negated, this bit is not used). Furthermore, if literal ℓ occurs in clause c_j, then the weight has a 1-digit in the position corresponding to clause c_j in the verification-part. All other digits and bits are 0.
- For every literal $\ell \in \{y_i, \overline{y_i}\}$ there is a corresponding Y-item $Y(\ell)$. Its weight has a 1-digit in the position corresponding to variable y_i in the Y-part. If literal ℓ occurs in clause c_j, then the weight of $Y(\ell)$ has a 1-digit in the position corresponding to clause c_j in its verification-part. All other digits and bits are 0.
- For every clause c_j there are three C-items $C^k(c_j)$ with $k = 0, 1, 2$. The weight of item $C^k(c_j)$ has a 1-digit in the position corresponding to clause c_j in the clause-part, and a k-digit in the position corresponding to clause c_j in the verification-part. All other digits and bits are 0.
- Finally there are two dummy items D_0 and D_1 whose weights are $w(D_0) = U - 1$ and $w(D_1) = U + 2^s$. The integer U in these weights is defined as follows: it has a 3-digit in every position in the verification-part, a 1-digit in every position in the clause-part, Y-part, and X-part, and an all-zero control-part.

We will throughout refer to the $4s + 3t$ non-dummy items as XYC-items. To complete the description of the Captive-Queen instance, we define the weight bound $\Delta = 1$, the initial state $I_0 = \{D_0\}$ with $w(I_0) = U - 1$, and the goal state $I_1 = \{D_1\}$ with $w(I_1) = U + 2^s$.

▶ **Lemma 6.** *The constructed instance of* Captive-Queen *satisfies the following.*

(i) *If we add up the decimal representations of the weights of some subset J of XYC-items, then there will be no carry-overs from lower positions to higher positions in*

the verification-part, clause-part, Y-part, and X-part. Furthermore, there will be no carry-over from the control-part to the X-part.

(ii) *If $w(I_0) < V < w(I_1)$ holds for some integer V, then the verification-part, clause-part, Y-part, and X-part of V agree with the corresponding part of U. The control-part of V lies between 0 and $2^s - 1$.*

(iii) *If $w(I_0) < w(J) < w(I_1)$ holds for some item set J, then J contains no dummy items and hence solely consists of XYC-items.*

Proof. Statement (i) follows by looking into the digits and bits in our construction. Only X-items $X(x_i)$ for un-negated literals have non-zero control-part, and these control-parts altogether only add up to $2^s - 1$. Every position in the decimal representation of verification-part, clause-part, Y-part, or X-part is non-zero for at most five XYC-items. For statement (ii), note that $U \leq V \leq U + 2^s - 1$ and note that the control-part of U is 0.

For statement (iii), observe that all item weights in the instance are greater than 10^s. If set J contains dummy item D_0 then it must also contain some other item, and this brings $w(J)$ above $w(I_1)$. And if J contains dummy item D_1 then its weight is above $w(I_1)$. ◀

We now define a bijection between the 2^s integers V with $U \leq V \leq U + 2^s - 1$ on one side and the 2^s truth-settings of the Boolean variables in $X = \{x_1, \ldots, x_s\}$ on the other side. Since $0 \leq V - U \leq 2^s - 1$, the binary representation of $V - U$ consists of s bits. Then the ith bit (counted from the right end) specifies the truth-value of variable x_i in the corresponding truth-setting $T_V(X)$. Vice versa, any truth-setting for X can be interpreted as the binary representation of some integer where the value of x_i specifies the ith bit. By adding the value U to this integer, we get the number from the range $U, \ldots, U + 2^s - 1$ that corresponds to the truth-setting.

The following two lemmas will be proved in the full version of this paper.

▶ **Lemma 7.** *Let V be an integer with $U \leq V \leq U + 2^s - 1$. If there exists an item set J with $w(J) = V$, then there also exists a truth-setting $T(Y)$ for the Boolean variables in Y, such that formula $\phi(X, Y)$ is true under the combined truth-setting $T_V(X)$ and $T(Y)$.*

▶ **Lemma 8.** *Let V be an integer with $U \leq V \leq U + 2^s - 1$. If there exists a truth-setting $T(Y)$ for the Boolean variables in Y such that formula $\phi(X, Y)$ is true under the combined truth-setting $T_V(X)$ and $T(Y)$, then there exists an item set J with $w(J) = V$.*

Let us wrap things up. Assume that the constructed instance of Captive-Queen has answer YES. By Corollary 4 this is the case if and only if for all integers V in the range $w(I_0) < V < w(I_1)$ there exists an item set J with $w(J) = V$. By Lemma 7 and Lemma 8 this is the case if and only if for all truth-settings $T_V(X)$ with $U \leq V \leq U + 2^s - 1$ for the variables in X, there exists a truth setting for the variables in Y such that formula $\phi(X, Y)$ is true. And finally this exactly means that the considered instance of 2-Quantified 3-CNF-Sat has answer YES. Together with Lemma 5 this yields the main result of the paper.

▶ **Theorem 9.** *Problem* Captive-Queen *is Π_2^p-complete.* ◀

Our reduction establishes Π_2^p-hardness of Captive-Queen for the special case where $\Delta = 1$. If we multiply all item weights in our construction by a factor f, then we also derive Π_2^p-hardness for the cases where $\Delta = f$.

Finally let us settle the complexity of Captive-Queen-with-few-Moves

▶ **Theorem 10.** *Problem* Captive-Queen-with-few-Moves *is NP-complete.*

Proof. The NP-certificate consists of the at most m intermediate states that the system traverses while moving from the initial state to the goal state.

The NP-hardness proof is done by a reduction from the NP-hard Subset-Sum problem (see Garey & Johnson [3]): Given a sequence $q_1, \ldots, q_n$ of positive integers and a positive integer Q, is there an index-set $N \subseteq \{1, \ldots, n\}$ with $q(N) = Q$? Consider the item set $I = \{1, \ldots, n+2\}$ with $w(i) = 2q_i$ for $1 \leq i \leq n$, and with $w(n+1) = 2Q - 1$ and $w(n+2) = 2Q + 1$. The difference bound is $\Delta = 1$, the initial state is $I_0 = \{n+1\}$, the goal state is $I_1 = \{n+2\}$, and the bound on the number of moves is $m = 2$. The only way of moving from I_0 to I_1 is through a state $N \subseteq \{1, \ldots, n\}$ with $q(N) = Q$. ◀

4 Two highly structured special cases

In this section we analyze special cases of Captive-Queen where the weight sequence carries a strong combinatorial structure and therefore behaves nicely. We will show that for these special cases the (otherwise difficult) problems Captive-Queen and Captive-Queen-with-few-Moves become solvable in polynomial time. The following two auxiliary tools T1 and T2 (for an item set I with weights $w(i)$ for $i \in I$) form the main ingredients for our algorithms.

T1. Compute the maximum gap in the weight spectrum between two given bounds $w(I_0)$ and $w(I_1)$; this maximum gap is denoted by $\text{gap}(I, w, I_0, I_1)$.
T2. Compute the largest value W with $W \leq d$ in the weight spectrum; the corresponding value is denoted $W_{\max}(I, w, d)$.

▶ **Lemma 11.** *Consider a specially structured family of weight sequences for which the tools T1 and T2 can be implemented in polynomial time. Then for this family also the problems* Captive-Queen *and* Captive-Queen-with-few-Moves *can be solved in polynomial time.*

Proof. By Corollary 4.(ii) an instance of Captive-Queen has answer YES if and only if $\text{gap}(I, w, I_0, I_1) \leq \Delta$. Furthermore, an instance of Captive-Queen-with-few-Moves can be solved as follows. We let $d_0 = w(I_0)$ and then compute the auxiliary values $d_j = W_{\max}(I, w, d_{j-1} + \Delta)$ for $j = 1, \ldots, m$. The instance has answer YES if and only if $d_m \geq w(I_1)$. ◀

4.1 The case with super-increasing weight sequences

In this section we consider item sets $I = \{1, \ldots, n\}$ whose weights are super-increasing and hence satisfy the following inequalities. These conditions originate from the knapsack literature; see for instance Magazine, Nemhauser & Trotter [5].

$$w(1) + w(2) + \cdots + w(i-1) \;<\; w(i) \qquad \text{for } i = 1, \ldots, n. \tag{3}$$

With every subset $J \subseteq I$ we associate a binary number $\text{bin}(J) = b_n b_{n-1} \ldots b_2 b_1$ whose bits are defined as $b_i = 1$ if $i \in J$ and $b_i = 0$ if $i \notin J$. Furthermore we denote by J^+ the subset with $\text{bin}(J^+) = \text{bin}(J) + 1$ (in case $J \neq I$), and we denote by J^- the subset with $\text{bin}(J^-) = \text{bin}(J) - 1$ (in case $J \neq \emptyset$). It is easy to see (and also well-known) that $w(J) < w(K)$ holds if and only if $\text{bin}(J) < \text{bin}(K)$. Hence distinct subsets always have distinct weights, and the weight spectrum consists of 2^n pairwise distinct values. For any set $J \subseteq I$ with $\emptyset \neq J \neq I$, the three numbers $w(J^-)$, $w(J)$, $w(J^+)$ form three consecutive values in the weight spectrum.

Now let us discuss the gap between two consecutive values $w(J)$ and $w(J^+)$ (with $J \neq I$) in the weight spectrum. Let $k \in I$ be the smallest element that is not contained in J. Since $\text{bin}(J^+) = \text{bin}(J) + 1$, the set J^+ results from J by adding element k to it while simultaneously removing the elements $1, 2, \ldots, k-1$ from it. This yields that the gap length $w(J^+) - w(J)$ equals

$$G_k \;:=\; w(k) - \sum_{j=1}^{k-1} w(j). \tag{4}$$

Next consider an input I, w, I_0, I_1 for tool T1. Let α be the largest element in the symmetric difference of I_0 and I_1; then $\alpha \notin I_0$ and $\alpha \in I_1$. Define an intermediate set $I_{1/2}$ that agrees with I_0 and I_1 on all elements above α, that contains α, and that contains none of the elements below α. From now on we assume that $I_0 \neq I_{1/2}^-$ and $I_1 \neq I_{1/2}$, as the cases with equality are easily settled. The maximum gap between $w(I_0)$ and $w(I_1)$ either is the maximum gap between $w(I_0)$ and $w(I_{1/2}^-)$, or the $w(I_{1/2}^-)$ and $w(I_{1/2})$, or it is the maximum gap between $w(I_{1/2})$ and $w(I_1)$. Hence it is sufficient to determine these three gaps, and then to output the value of the largest one.

In order to analyze the first gap, let β be the largest element with $\beta \notin I_0$ that is strictly smaller than α. Since $I_{1/2}^-$ does contain β and also all the elements below β, the maximum gap between $w(I_0)$ and $w(I_{1/2}^-)$ equals $\max_{k \leq \beta} G_k$. The second gap between $w(I_{1/2}^-)$ and $w(I_{1/2})$ equals G_α. For the third gap, let γ be the largest element with $\gamma \in I_1$ that is strictly smaller than α. Since $I_{1/2}$ neither contains γ nor any of the elements below γ, the maximum gap between $w(I_{1/2})$ and $w(I_1)$ equals $\max_{k \leq \gamma} G_k$. This completes the polynomial time algorithm for tool T1.

A polynomial time algorithm for tool T2 can be found in the literature (Magazine, Nemhauser & Trotter [5]), and is based on a simple greedy approach. Consider a knapsack of size d, and repeatedly pack the largest unpacked item weight into this knapsack. When no further item fits into the knapsack, the overall weight in the knapsack equals $W_{\max}(I, w, d)$.

▶ **Theorem 12.** *Problems* Captive-Queen *and* Captive-Queen-with-few-Moves *can be solved in polynomial time, if the weight sequence is super-increasing.* ◀

4.2 The case with divisible weight sequences

In this section we consider item sets $I = \{1, \ldots, n\}$ whose weights satisfy the following divisibility conditions. These conditions come from the knapsack and packing literature where they have been investigated thoroughly; see for instance Pochet & Wolsey [8] and Coffman, Garey & Johnson [1].

$$w(i) \mid w(i+1) \qquad \text{for } i = 1, \ldots, n-1. \tag{5}$$

Our first goal is to design a polynomial time algorithm for tool T1. We distinguish two cases. First assume that $w(1) > 1$. Then (5) implies that all item weights are divisible by $w(1)$. In this case we define new item weights $w'(i) = w(i)/w(1)$, and observe that

$$\text{gap}(I, w, I_0, I_1) \;=\; w(1) \cdot \text{gap}(I, w', I_0, I_1). \tag{6}$$

Next assume that $w(1) = 1$ holds, and define ℓ as the largest integer with $w(\ell) = 1$. Let $I' = \{\ell + 1, \ldots, n\}$ contain the items of weight greater than 1, let w' be the restriction of the weights w from I to I', and let $I_0' = I_0 \cap I'$ and $I_1' = I_1 \cap I'$. Then

$$\text{gap}(I, w, I_0, I_1) \;=\; \max\left\{\text{gap}(I', w', I_0', I_1') - \ell,\; 1\right\}. \tag{7}$$

Note that $I' = \emptyset$ in (7) yields $\text{gap}(I, w, I_0, I_1) = 1$. The two formulas in (6) and (7) yield a recursive procedure for computing $\text{gap}(I, w, I_0, I_1)$. The running time of the procedure is polynomial, and with a little effort can even be made linear in n.

Tool T2 is available in the literature (Coffman, Garey & Johnson [1]), and follows the same greedy approach as tool T2 for super-increasing weight sequences.

▶ **Theorem 13.** *Problems* Captive-Queen *and* Captive-Queen-with-few-Moves *can be solved in polynomial time, if the weight sequence is divisible.* ◀

5 Analysis of four related problems

We will now discuss some further properties of the discrete system that underlies the Captive-Queen problem. Recall that every state of the system corresponds to a subset of items, and that the system can move from state J directly to state K if (1) respectively (2) is satisfied. A state J is *isolated*, if there are no other states reachable from J. A system is *fully connected*, if every state is reachable from every other state. Equivalently, a system is fully connected if and only if the state $\emptyset$ is reachable from the state I.

Figure 3 lists four algorithmic problems that are formulated around isolated states and fully connected systems. The full version of this paper will show that

- Problem Isolated-State is coNP-complete;
- Problem All-States-Isolated is coNP-complete;
- Problem Some-State-Isolated can be decided in polynomial time;
- Problem Fully-Connected can be solved in polynomial time.

Furthermore, the full version will show that all these problems are easy, if the weight sequence is super-increasing (see Section 4.1) or divisible (see Section 4.2).

Problem: Isolated-State
Instance: An item set I; weights $w(i)$ for $i \in I$; a bound Δ; a subset $J \subseteq I$.
Question: Is the state J isolated?

Problem: All-States-Isolated
Instance: An item set I; weights $w(i)$ for $i \in I$; a bound Δ.
Question: Are all states in this system isolated?

Problem: Some-State-Isolated
Instance: An item set I; weights $w(i)$ for $i \in I$; a bound Δ.
Question: Does this system contain some isolated state?

Problem: Fully-Connected
Instance: An item set I; weights $w(i)$ for $i \in I$; a bound Δ.
Question: Is this system fully connected?

Figure 3 The algorithmic problems discussed in Section 5.

References

1 E.G. Coffman Jr., M.R. Garey, and D.S. Johnson (1987). Bin packing with divisible item sizes. *Journal of Complexity 3*, 406–428.

2 T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein (2001). *Introduction to Algorithms*. MIT Press.

3 M.R. Garey and D.S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco.

4 H. Kellerer, U. Pferschy, and D. Pisinger (2004). *Knapsack problems*. Springer Verlag, Berlin.

5 M. Magazine, G.L. Nemhauser, and L.E. Trotter (1975). When the greedy solution solves a class of knapsack problems. *Operations Research 23*, 207–217.

6 S. Martello and P. Toth (1990). *Knapsack problems: Algorithms and computer implementations*. John Wiley & Sons, Chichester.

7 C.H. Papadimitriou(1994). *Computational Complexity*. Addison-Wesley.

8 Y. Pochet and L.A. Wolsey (1995). Integer knapsack and flow covers with divisible coefficients: Polyhedra, optimization and separation. *Discrete Applied Mathematics 59*, 57–74.

9 L.J. Stockmeyer (1977). The polynomial-time hierarchy. *Theoretical Computer Science 3*, 1–22.

10 E. Wakeling (1995). *Rediscovered Lewis Carroll puzzles*. Courier Dover Publications.

11 G.J. Woeginger and Z. Yu (1992). On the equal-subset-sum problem. *Information Processing Letters 42*, 299–302.